IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re: **Nadalin et al.**	§	Group Art Unit: **2132**
	§	
Serial No.: **09/321,788**	§	Examiner: **Kim, J.**
	§	
Filed: **05/27/1999**	§	Attorney Docket #: **AT9-99-081**
	§	
For: **Method for enabling a program written in untrusted code to interact with a security subsystem of a hosting operating system**		

APPELLANT'S BRIEF
IN RESPONSE TO OFFICE ACTION UNDER 37 C.F.R. § 41.37

This brief is filed in support of the Notice of Appeal, filed xx/xx/xxxx, and which appeals the rejection of claims 11-28 from the decision of the examiner dated xx/xx/xxxx.

I. REAL PARTY IN INTEREST

The real party in interest in this appeal is International Business Machines Corporation (IBM).

II. RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

III. STATUS OF CLAIMS

Claims 11-28 are pending in this application; claims 11-28 have been finally rejected; and claims 11-28 have been appealed. Claims 1-10 have been canceled. No claims have been allowed or withdrawn.

IV. STATUS OF AMENDMENTS

No after-final amendments have been filed.

V. SUMMARY OF CLAIMED SUBJECT MATTER

A program written in untrusted code (e.g., JAVA) is enabled to access a native operating system resource (e.g., supported in WINDOWS NT) through a staged login protocol. In operation, a trusted login service listens, e.g., on a named pipe, for requests for login credentials (Original specification, page 11, line 5; step 78, FIG. 2). In response to a login request, the trusted login service requests a native operating system identifier (page 11, line 19; step 83, FIG. 3). The native operating system identifier is then sent to the program (page 12, line 1; step 89, FIG. 3). Using this identifier, a credential object is then created within an authentication framework (page 12, line 21; step 96, FIG. 2). The credential object is then used to login to the native operating system to enable the program to access the resource (page 13, line 2; step 38, FIG. 1). This technique enables a JAVA program to access a WINDOWS NT operating system resource under the identity of the user running the JAVA program (page 15, line 16).

VI. Grounds of rejection to be reviewed on appeal

The grounds of rejection that are on appeal are:

(A) whether claims 24-28 are incomplete for omitting essential steps under 35 U.S.C. § 112; and

(B) whether claims 11-23 are unpatentable under 35 U.S.C. § 103(a) over Stallings, *Cryptography and Network Security: Principles and Practice*, in view of Bittinger et al., "Systems, methods and computer program products for invoking server applications using tickets registered in client-side remote object registries", U.S. Patent Number 6,453,362, filed 08/12/1998, issued 09/17/2002, and further in view of Praun, "Moving UNIX Applications to Windows NT", and further in view of Hunt, *TCP/IP Network Administration*.

VII. ARGUMENTS

VII.A. Was 35 U.S.C. § 112 properly applied in a rejection of claims 24-28 as being incomplete for omitting essential steps?

Claims 24-28 stand and fall together as a single group.

Independent claim 24 was a new claim in the response to the non-final Office action, and this rejection was newly introduced in the final Office action. Appellant notes that the final Office action rejected only independent claim 24 under 35 U.S.C. § 112; however, since dependent claims 25-28 depend from claim 24, the dependent claims incorporate any supposed deficiency of the independent claim under 35 U.S.C. § 112 and, therefore, should have also been similarly rejected.

The rejection states in its entirety (emphasis in original):

> Claim 24 is rejected under 35 U.S.C. § 112, second paragraph, as being incomplete for omitting essential steps, such omission amounting to a gap between the steps. See MPEP § 2172.01. The omitted steps are: using the credential object by the program to access the resource within the native operating system by impersonating a security context of a native operating system user in accordance with the credential object (claim 24, last step; see specification, page 13, lines 19-24). This step is essential in the method to distinguish the step of '[accessing] in a trusted manner' as claimed in the preamble.

Claim 24 already included the feature of "using the credential object by the program to access the resource within the native operating system" as its last step. Hence, the rejection argues that the feature in the emphasized text, i.e. "by impersonating a security context of a native operating system user in accordance with the credential object", should have been included within the claim because of the wording of the preamble. Appellant disagrees.

There is no valid reason for requiring the additional feature in claim 24 as argued in the rejection. The preamble of independent of claim 24 reads as follows: "A method for enabling a program written in untrusted code to access in a trusted manner a resource supported on a computing device executing a native operating system, the method comprising: ...". More succinctly, with respect to the argument in the rejection, the preamble states that a resource is accessed in a trusted manner. Given that the usage of credentials are well-known in trust operations when accessing computational resources, Appellant asserts that one having ordinary skill in the art would agree that using a credential to access a resource is one of many possible ways of accessing a resource in a trusted manner. Thus, Appellant asserts that "using the credential object ... to access the resource

within the native operating system" is equivalent to the stated purpose within the preamble. Therefore, claim 24 does not lack any essential steps.

For this and other reasons, Appellant argues that the position of the Examiner should be reversed and that the rejection of claims 24-28 under 35 U.S.C. § 112 should not be upheld.

VII.B. Was 35 U.S.C. § 103(a) properly applied in a rejection of claims 11-23 as being unpatentable over Stallings, Bittinger et al., Praun, and Hunt?

Claims 11-23 stand and fall together as a single group.

Arguments in support of separate patentability

As an initial point, Appellant notes that the rejection states that only claims 11-13 and 17-19 are rejected over Stallings, Bittinger et al., Praun, and Hunt. However, the argument within the rejection is also applied against claims 21-23. Thus, the rejection should have stated that claims 21-23 were also similarly rejected.

As another initial point, Appellant notes that claims 14-16 and 20 were rejected over Stallings, Bittinger et al., Praun, and Hunt and further in view of additional prior art references. Given that dependent claims 14-16 depend from independent claim 11, and given that dependent claim 20 depends from independent claim 17, Appellant has not separately argued for the patentability of claims 14-16 and 20 and merely argues in support of the independent claims from which these claims depend.

With respect to separate patentability, Appellant asserts that independent 17 is the broadest claim in the patent

application. Hence, for purposes of this argument, Appellant argues for the patentability of claims 11-23 of the present invention using claim 17 as an exemplary claim. The rejection applies an obviousness argument against independent claim 17, and then the rejection states that independent claims 11 and 21 are similar to claim 17 and are therefore rejected using the argument that is used against independent claim 17. Appellant's argument hereinbelow with respect to the non-obviousness of claim 17 is also applicable to independent claims 11 and 21 because each of these independent claims also includes a similar feature by reciting claim language that is similar to "wherein the login request contains an identifier for a uniquely-named response pipe".

Argument supporting the present claims in view of the rejection over *Stallings, Bittinger et al., Praun,* and *Hunt*

Independent claim 17 reads as follows:

17. A computer program product in a computer readable medium for enabling a program written in untrusted code to access a native operating system resource, the computer program product comprising the steps of:

means for listening on a named pipe by a trusted login service for login requests;

means responsive to a login request for requesting a native operating system identifier by the trusted login service, wherein the login request contains an identifier for a uniquely-named response pipe;

means for returning to the program via the uniquely-named response pipe the native operating system identifier, wherein the uniquely-named response pipe and the named pipe are not identical;

in an authentication framework, using the native operating system identifier to create a credential object; and

using the credential object to login to the native operating system to enable the program to access the resource.

The argument in the rejection states that Stallings discloses various claimed features. In particular, the rejection states in paragraph 8 on pages 3 and 4 of the final Office action that Stallings discloses the following features:

a. listening by a trusted login server for login requests ...

b. responsive to a login request for requesting a native operating system identifier by the trusted login server ...

c. returning to the program the native operating system identifier ...

d. in an authentication framework, using the native operating system identifier to create a credential object ...

e. using the credential object to login to the server to enable the program to access the resource ...

Appellant assumes *arguendo* that the rejection: (a) has fairly characterized the teachings of Stallings and (b) has accurately applied the disclosure of Stallings against claim 17.

The rejection further states on page 4, paragraph 9, of the final Office action that the authentication dialogue that is disclosed by Stallings is "a service between a user and a trusted operating system and not between untrusted code and a native operating system". The rejection then states that "'users' of a secure OS generalize to include other types of relations such as an unprivileged client application gaining access to a secured server application", and the rejection argues that this feature is taught by Bittinger et al.. Appellant assumes *arguendo* that the rejection: (a) has fairly characterized the teachings of Bittinger et al.; (b) has accurately applied the disclosure of Bittinger et al. against claim 17; and (c) has properly provided a motivational statement for combining the teachings of Stallings and Bittinger et al..

The rejection further states in paragraph 10 on page 5 of the final Office action that "Stallings is silent on the matter of establishing communication between the program and the trusted login server using named pipes". The rejection notes that Praun teaches the use of "ImpersonateNamedPipeClient" within Windows NT for acquiring privileges in a security context. Appellant assumes *arguendo* that the rejection: (a) has fairly characterized the teachings of Praun; (b) has accurately applied the disclosure of Praun against claim 17; and (c) has properly provided a motivational statement for combining the teachings of Stallings and Bittinger et al. with Praun.

The rejection continues in paragraph 11 on pages 5-6 of the final Office action by noting that Stallings fails to teach a claimed feature:

> Stallings is silent on the matter of the login request containing an identifier for a uniquely-named response pipe, wherein the uniquely-named response pipe and the named pipe on which the trusted login service is listening for login requests are not identical.

The rejection then continues by explaining a feature that is taught by Hunt:

> However, in the analogous art of TCP handshaking, Hunt teaches a means wherein a source generates a distinct pair (source port, destination port) and exchanges this pair to the destination: the destination using the source port value and destination port value received from the source as its destination port and source port respectively (see Hunt, page 49, Figure 2.7). In this scheme, the source transmits messages to the destination on the source port and listens on the destination port for a response by the destination.

The rejection then provides a motivational statement for combining Stallings and Hunt:

> It would have been obvious to one of ordinary skill in the art at the time the invention was made to apply the teaching of Hunt to the invention covered by Stallings since this type of configuration is a typical means for a system to dynamically connect with multiple clients as taught by Hunt (see Hunt, page 48, last paragraph-page 49, first paragraph).

Appellant assumes *arguendo* that the rejection has fairly characterized the teachings of Hunt.

However, the teachings of Hunt have not been properly applied to the claimed features as implied by the rejection. In fact, the rejection follows the form of a bait-and-switch.

The rejection argues that Hunt teaches two features in the claims for which Stallings is silent (using the language of the rejection): (feature 1) "the matter of the login request containing an identifier for a uniquely-named response pipe" and (feature 2) "the matter of ... [feature 1 and] wherein the uniquely-named response pipe and the named pipe on which the trusted login service is listening for login requests are not identical." These are aspects of two elements in claim 17, which recites:

> ...
> means responsive to a login request for requesting a native operating system identifier by the trusted login service, wherein the login request contains an identifier for a uniquely-named response pipe;
> means for returning to the program via the uniquely-named response pipe the native operating system identifier, wherein the uniquely-named response pipe and the named pipe are not identical; ...

Appellant argues that the rejection has failed, at a minimum, to argue or to explain how Hunt teaches the first feature, i.e. "the matter of the login request containing an identifier for a uniquely-named response pipe".

Appellant assumes *arguendo* that the rejection has fairly applied the teachings of Hunt of a source port and a destination port with respect to the claimed feature of a response pipe and a named pipe that are not identical. In other words, Hunt may teach the usage of uniquely-identified ports and the usage of different ports for sending data and receiving data, and this might be analogous to the response pipe and the named pipe in the claims. Hence, the argument in the rejection with respect to the second feature may be valid.

However, even if the teachings of Hunt are fairly applied with respect to the usage of named pipes, the rejection has not attempted to address how the teachings of Hunt or any other applied prior art reference disclose the feature of including an identifier for a uniquely-named response pipe within the login request. There is no suggestion of including such information within the login request; in other words, the rejection fails to explain how the prior art discloses the first feature that is mentioned hereinabove on the previous page of this appeal brief.

Appellant asserts that the rejection has improperly followed the claimed elements of the present invention as a roadmap to piecing together the features in the prior art, or in other words, that the Examiner has used improper hindsight in forming the rejection. In fact, both Stallings and Hunt teach away from the present invention. The methodology that is taught by Stallings assumes that the communication path has already been created; Kerberos tickets, etc., are exchanged over a communication path that already exists. Moreover, Stallings teaches security operations that are well-known; it is unclear how one would use the well-known security operations that are taught by Stallings if one were required to customize the methodology to include the creation of the communication path in

addition to the exchange of Kerberos information. Furthermore, the teachings of Hunt that are applied against the claims show that ports are designated in a very specific way during the TCP handshake, and it is unclear how one would move that information into a login request message without destroying the port/socket mechanism. For example, the TCP handshake would have had to have occurred before the exchange of information higher within the network protocol stack, i.e. the exchange of messages in the application layer.

Rejections are deficient with respect to requirements for a proper obviousness rejection

With respect to claims 11-23 of the present patent application, Appellant respectfully submits that the applied references cannot be combined to produce the claimed invention. Hence, a rejection of claims 11-23 cannot be based upon the cited prior art to establish a *prima facie* case of obviousness. Therefore, a rejection of the claims under 35 U.S.C. § 103(a) has been shown to be improper and insupportable in view of the cited prior art, and claims 11-23 are patentable over the applied references. For this and other reasons, Appellant argues that the position of the Examiner should be reversed and that the rejection of claims 11-23 should not be upheld.

VIII. APPENDIX OF CLAIMS

1. (Canceled).

2. (Canceled).

3. (Canceled).

4. (Canceled).

5. (Canceled).

6. (Canceled).

7. (Canceled).

8. (Canceled).

9. (Canceled).

10. (Canceled).

11. A method for enabling a program written in untrusted code to access a native operating system resource, comprising the steps of:

having a trusted login service listen on a named pipe for login requests;

responsive to a login request, wherein the login request contains an identifier for a uniquely-named response pipe, having the trusted login service request a native operating system identifier;

returning to the program via the uniquely-named response pipe the native operating system identifier, wherein the uniquely-named response pipe and the named pipe are not identical;

in an authentication framework, using the native operating system identifier to create a credential object; and

using the credential object to login to the native operating system to enable the program to access the resource.

12. The method as described in claim 11 wherein the native operating system supports named-pipe servers.

13. The method as described in claim 12 wherein the program is written in an interpreted language.

14. The method as described in claim 11 wherein the authentication framework is a pluggable authentication mechanism (PAM) having a set of application programming interfaces (APIs).

15. The method as described in claim 14 wherein the set of application programming interfaces include login, commit, abort and logout APIs.

16. The method as described in claim 14 wherein the authentication framework is compliant with an authentication service of a virtual machine.

17. A computer program product in a computer readable medium for enabling a program written in untrusted code to access a native operating system resource, the computer program product comprising the steps of:

means for listening on a named pipe by a trusted login service for login requests;

means responsive to a login request for requesting a native operating system identifier by the trusted login service, wherein the login request contains an identifier for a uniquely-named response pipe;

means for returning to the program via the uniquely-named response pipe the native operating system identifier, wherein the uniquely-named response pipe and the named pipe are not identical;

in an authentication framework, using the native operating system identifier to create a credential object; and

using the credential object to login to the native operating system to enable the program to access the resource.

18. The computer program product as described in claim 17 wherein the program executes in a virtual machine supported by the native operating system and the native operating system supports named-pipe servers.

19. The computer program product as described in claim 17 wherein the program is written in an interpreted language.

20. The computer program product as described in claim 17 wherein the authentication framework is compliant with an authentication service of a virtual machine.

21. An application server, comprising:

a set of programs that are supported by a virtual machine that is supported by a native operating system;

a processor running the native operating system providing support for executing the set of programs; and

means for enabling each program in the set of programs to run in an operating system thread while impersonating a different native operating system user in accordance with a token that was created during a login operation in the native operating system and that was associated with a program while the program was acting as a named-pipe server to listen for a login response on a named pipe that was uniquely created for a login request to obtain the token, wherein the login request contained an identifier for the named pipe.

22. The application server as described in claim 21 wherein the native operating system supports named-pipe servers.

23. The application server as described in claim 21 further including a server application executed by the processor for receiving a request for service from a client machine and initiating execution of a program in the set of programs in a given operating system thread.

24. A method for enabling a program written in untrusted code to access in a trusted manner a resource supported on a computing device executing a native operating system, the method comprising:

listening, by a trusted login service in the native operating system, for login requests on a named request pipe;

generating a login request at the program, wherein the login request contains authentication information and an identifier for a named response pipe, wherein the named request pipe and the named response pipe are not identical;

in response to creating the named response pipe by the program, acting as a named-pipe server on the named response pipe by the program;

in response to receiving the login request on the named request pipe at the trusted login service from the program, performing a login operation with the authentication information by the trusted login service into the native operating system;

in response to performing the login operation, sending a login response on the named response pipe from the trusted login service to the program;

in response to receiving the login response on the named response pipe at the program from the trusted login service, closing the named response pipe such that the named response pipe is uniquely associated with the login request and is not used for additional login requests;

in response to receiving the login response on the named response pipe at the program from the trusted login service, creating a credential object by the program using a token generated during the login operation; and

using the credential object by the program to access the resource within the native operating system.

25. The method as described in claim 24 further comprising:

prior to sending the login response but after performing the login operation, performing a first impersonation operation by the trusted login service, wherein the first impersonation operation is based on the token such that the trusted login service impersonates a security context of a user associated with the authentication information; and

prior to closing the named response pipe but after receiving the login response, performing a second impersonation operation by the program, wherein the second impersonation operation is based on the trusted login service acting as a named-pipe client on the named response pipe and the program acting as the named-pipe server on the named response pipe such that the program impersonates a security context of the login response as a most recent message read from the named response pipe.

26. The method as described in claim 25 further comprising:

after sending the login response, performing a first revert operation by the trusted login service to terminate its previous impersonation; and

after performing a token duplication operation by the program, performing a second revert operation by the program to terminate its previous impersonation.

27. The method as described in claim 25 further comprising:

performing a token duplication operation by the program, wherein the token duplication operation associates the token with a thread that initiated a login in order to obtain access to the resource.

28. The method as described in claim 27 further comprising:

after performing the token duplication operation, performing a third impersonation operation by the program, wherein the third impersonation operation is based on the credential object such that a calling thread impersonates a security context of a user associated with the authentication information.

IX. Evidence appendix

None.

X. Related proceedings appendix

None.

XI. Conclusion

In view of the above arguments, it is respectfully urged that the rejection of the claims should not be sustained.

DATE: 10/24/06

Respectfully submitted,

Jeffrey S. LaBaw
Reg. No. 31,633
Attorney
IBM Intellectual Property Law
Austin, Texas 78758